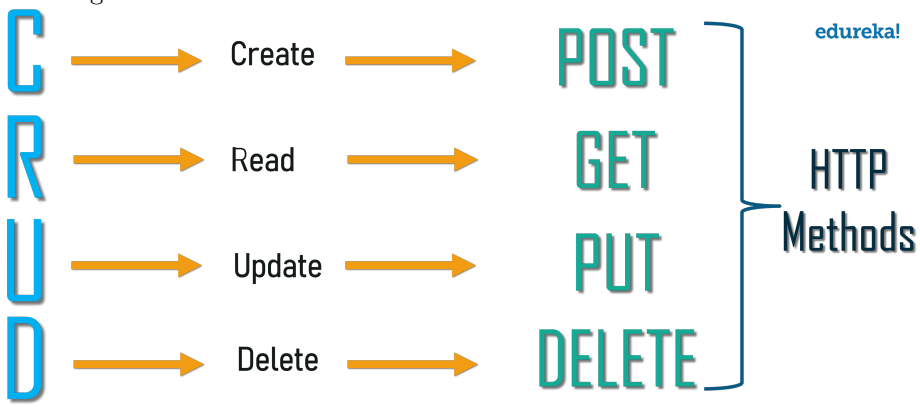
1 Wie tauschen wir Daten zwischen unserem Client und unserem Server aus?

Wir benutzen eine RestAPI um über das HTTP-Protokol Daten zwischen unserem Server und unserem Client auszutauschen.

In einer RestAPI werden die CRUD-Operationen durch HTTP-Request Methoden dargestellt.



1.1 Beispiel Benutzer erstellen

Es werden an verschiedene API-Endpunkte Anfragen gestellt um Daten auf dem Server zu verwalten. Um beispielsweise einen Benutzer zu erstellen, wird ein POST-Request an den API-Endpunkt /users gesendet.

```
POST: [api-url]/users
```

Damit ein neuer Nutzer erstellt werden kann, werden weitere informationen benötigt. Diese können entweder als Parameter in der URL eingefügt werden oder im JSON-Format als Payload mitgeschickt werden.

1.1.1 Erstellen mit URL-Parametern

Zum erstellen eines Nutzers sind beispielsweise ein Benutzername und ein Passwort nötig. Diese können als Parameter in der URL mitgeschickt werden. Parameter werden an das Ende der URL eingefügt und beginnen hinter einem ?. Jeder weitere Parameter kann mit einem &-Zeichen angehängt werden. In Beispiel des erstellen eines Benutzers könnte eine Anfrage dann so aussehen:

```
POST: [api-url]/users?username=max_musterman&password=bad_password
```

1.1.2 Erstellen mit JSON-Payload

Alternativ kann auch ein JSON-Objekt mit an den Server gesendet werden. Die URL bleibt dann bei [api-url]/users aber es wird der HTTP-Anfrage eine JSON-Objekt mitgegeben, welches in unserem Fall so aussehen könnte:

```
{
    "username": "max_musterman",
    "password": "password"
}
```

1.1.3 Server Antwort

Der Server muss auf diese Anfragen beantworten. Als erstes gibt es einen Antwortcode welcher grundsätzlich Auskunft darüber gibt ob die Anfrage erfolgreich beantwortet werden konnte. In unsere Beispiel würde der Code 201 über das erfolgreiche erstellen eines neuen Benutzers informieren. Alle möglichen Codes sind hier von Mozilla aufgelistet. Zusätzlich kann, wie bei der Anfrage auch, über JSON Informationen zurückgegeben werden.

1.2 Authentication

Damit ein Benutzer nur seine Daten verändern kann, muss sichergestellt werden, dass der Nutzer auch das Recht hat die HTTP-Anfrage zu stellen die er stellt. Damit ein Benutzer beweisen kann, dass er das Recht hat seine Daten zu verändern, wird bei der Anfrage an den Server neben der URL und der Payload sogenannte HTTP-Header gesetzt. Wir benutzen diese Header um einen String mitzuschicken, mit welchem wir beweisen, dass wir der Nutzer sind, welcher wir behaupten zu sein.

Diesen String bekommt man, indem man sich mit Benutzername und Passwort anmeldet. Dieser String muss vom Client gespeichert werden, und wird jedes mal mitgeschickt, wenn eine Anfrage an den Server gestellt wird.

2 Code Beispiel

//TODO

3 API Dokumentation

Standart HTTP-Codes:

```
200: OK
201: Created
400: Bad Request
401: Unauthorized
403: Forbidden
500: Internal Server Error
501: Not implemented
```

3.1 Benutzer

3.1.1 Benutzer erstellen

Erstelle einen neuen Benutzer:

```
POST: [api-url]/users?username=[username]&password=[password]
```

Antwort:

```
200: Im Antwort Körper befindet sich die Antwort
```

JSON-Body Möglichkeiten:

1. Benutzer erfolgreich erstellt:

```
{
    "message": "user created",
    "user-id": "[user-id]",
    "token": "[jwt-token]"
}
```

2. Benutzer existiert schon:

```
{
    "message": "username taken"
}
```

3.1.2 Benutzer löschen

Lösche einen Benutzer:

```
DELETE: [api-url]/users/[user-id]
Header:
    Authorization: Bearer [jwt-token]
```

Antwort:

```
200: Benutzer wurde gelöscht
401: Nicht berechtigt Benutzer zu löschen (falsches jwt-token)
```